Sistemes Gràfics Interactius

CONTINGUT

1 SISTEMES DE MODELATGE GEOMÈTRIC

El modelatge geomètric estudia els principals esquemes de representació per a objectes 3D (sòlids, superficies, volums, fractals) així com les principals interrogacions i operacions geomètriques. SGI és continuació de VIG i és complementària de VA: A SGI s'estudia la construcció i modificació dels models geomètrics que podran ser visualitzats amb les tècniques que s'exposen a VA.

1.1 Estructura d'un Sistema de Modelatge Geomètric

El nucli de tot Sistema de Modelatge Geomètric (SMG) és el model geomètric 3D de l'objecte o objectes representats. El SMG ha d'incloure a més,

- Entrada interactiva (construcció de nous models)
- Transformació a d'altres models i conversió desde altres models
    - Operacions que no afecten a la forma: Transformacions geomètriques
    - Operacions que afecten a la forma: Operacions bool, seccions planes
- Interrogació: Generació d'informació no gràfica
    - Dimensions, angles
    - Area de la superficie de l'objecte
    - Volum
    - Centre de mases, moments d'inèrcia, etc.
- Visualització:
    - Filferros
    - Visualitzacions realistes
    - Visualitzacions NO realists

1.1.1 Operacions i Interrogacions bàsiques suportades pels SMG

Eines d'accés	**Interrogacions / Operacions**
Geometría dels vèrtexs 3D	Projeccions 3D Transformacions geomètriques
Incidències (interseccions de rectes) - Interseccio (recta, objecte) - Interseccio (Pla, objecte)	Eliminació de parts amagades Realisme, il.luminació Interrogacions
Classificacions - Classifica (Punt, Objecte) - Classifica (Recta, Objecte)	Operacions booleanes (unió, intersecció, diferència) Càlculs volumètrics
Propietats de punts	Isosuperfícies Segmentació

1.2 Models de Filferros, Superfícies, Sòlids i Volum

Els models es diferencien per la complexitat i volum d'informació geomèrica que emmagatzemen, i per les eines d'accés que poden suportar:

	Geometría 3D	**Incidències**	**Classificacions**	**Prop (Punts)**
Filferros	X			
Superficies	X	X		
Sòlids	X	X	X	
Superf. Avançades	X	X	X	
Models de Volum	X	X	X	X

Cal triar el model més adequat segons les necessitats de l'aplicació concreta. No és convenient usar models més complexes de l'imprescindible.

1.2.1 Característiques dels models

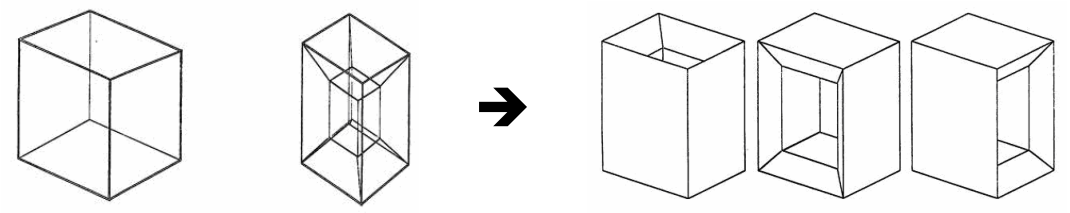
Ha d'existir una relació entre el model 3D representat al nostre SMG i l'objecte físic. Un cop especificat el conjunt d'operacions i interrogacions que requereix la nostra aplicació, el model s'espera que doni les mateixes respostes que obtindriem en la interrogació de l'objecte físic.

- **No ambigüitat:** Per tot model, no pot existir més d'un objecte real que tingui aquest model.
- **Validesa:** Sempre hi ha com a mínim un objecte real que correspon a aquest model.
- **Unicitat:** Per tot objecte real, li correspon un únic model
- **Concissió:** Quantitat de memòria requerida

1.2.2 Representació per filferros

- Pèrdua d'informació: Objecte, Model, Representació
- Representació: Llista d'arestes
    - Aresta: dos punters o indexs als seus vèrtexs extrems
    - Vèrtex: les seves coordenades (x,y,z)

Es un model no ambigu per a representar objectes fets realment amb estructures unidimensionals (per exemple filferros), pero és ambigu quan es vol usar per al modelatge de sòlids. Veurem un exemple de models en filferros, un dels quals pot donar lloc a tres objectes reals diferents, amb diferents propietats geometriques (volum, etc):



2 ESQUEMES DE REPRESENTACIÓ DE SÒLIDS. ALTRES MODELS

En aquest apartat parlarem de la representació d'objectes sòlids. Un sòlid és un objecte **finit** envoltat d'una **superfície orientada i tancada**, en el qual podem parlar de dins i fora. Els punts de la superfície d'un sòlid han de separar punts de dins dels punts de fora, no hi pot haver punts de la superfície que no tinguin punts interiors en el seu entorn (en altres paraules, no podem tenir cares o arestes aillades); això ho indicarem dient que el sòlid ha de ser **regular**.

En aquest curs considerarem a més que els nostres sòlids tenen totes les **cares planes**, i que són **dos-varietat** (l'entorn de qualsevol punt de la seva superfície es homeomorf a un disc).

2.1 Models de sòlids: propietats dels sòlids rígids

El fet de poder parlar de dins i fora permet definir l'operació bàsica de classificació. Donat un sòlid A, es pot parlar de la classificació de punts i rectes respecte S. En la **classificació de punts** donat un sòlid S i un punt P, la classificació de P respecte a S és una operació que ens retorna dins, sobre o fora en funció de si el punt P és a l'interior de S, sobre la seva superfície o bé és a l'exterior del sòlid S. De la mateixa manera, la classificació d'una recta r respecte S ens diu quins intèrvals de la recta són dins, sobre o fora del sòlid.

Hem dit també que la superfície del sòlid ha de ser orientada i tancada. **Orientada** vol dir que la superfície té dues bandes (la banda exterior que dona a fora de l'objecte i la banda interior que mira cap al seu interior) i que donat un punt qualsevol de la superfície Sup(S) sabem a quina banda té els punts interiors a S i a quina té els punts exteriors a S. **Tancada** vol dir que no és possible anar de cap punt interior a S a cap punt exterior a S sense passar per algun punt de la superfície SupS. El fet que la superfície sigui orientada i tancada, junt amb el requeriment de que no s'autointersecti, ens assegura que el sòlid queda determinat per la seva superfície. Podem "caminar" per la banda exterior de Sup(S) i sempre estarem a l'exterior de l'objecte; si caminem per la seva banda interior, ens trobarem sempre a l'interior de l'objecte. Coneixent la superfície del sòlid tenim tota la informació geomètrica del sòlid i podem per exemple classificar punts i rectes.

Com que la superfície del sòlid és el lloc geomètric dels punts que separen l'interior de l'exterior, també l'anomenarem frontera del sólid. A l'apartat següent presentarem la representació per fronteres, que ens permet emmagatzemar de forma no ambigua tota la informació geomètrica d'un sòlid guardant únicament informació de les cares de la seva frontera.

2.1.1 Políedres

Un poliedre es un sòlid limitat per cares planes. Com veurem al seguent apartat, els poliedres tenen cares, les cares tenen un o mes poligons, cada poligon es un cicle de tres o mes arestes i cada aresta te dos vertexs que son els seus extrems.

Per a que un poliedre sigui **tancat i dos varietat**, cal que totes les arestes siguin contigues a dues i només a dues cares. Si a més es cumpleix l'equació de Euler (veure seguent apartat) i no hi ha intersecció entre cares de la seva superfície (excepte en les arestes que puguin compartir), es pot assegurar que es cumpleixen les condicions que hem indicat més amunt i que és finit, tancat, orientable i regular. No obstant, i per a facilitar els algorismes, demanarem a més que les arestes exteriors de tota cara estiguin orientades cíclicament en el sentit del vector normal cap enfora del sòlid segons la regla del tirabuixó.

En resum: donada una superficie poliedrica (amb cares planes, poligons, arestes i vertexs), si cumpleix que cada aresta es compartida per dues i només dues cares, cumpleix a mes l'equació de Euler (veure seguent apartat) i que no hi ha intersecció entre cares, direm que aquesta superficie determina un sòlid dels que estudiem aquest curs. En cas contrari, estarem parlant d'un model de superficie pero no de sòlid.

2.1.2 Equació d'Euler

Si suposem que definim la frontera del sòlid mitjançant una representació explícita amb tres taules amb les cares, arestes i vèrtexs, en tot objecte sòlid es cumpleix l' Equació d'Euler:

Cares + Vèrtexs = Arestes + Anells + 2 * (Components - Forats)

C + V = A + R + 2 * (S - H)

El nombre de cares de l'objecte (C) més el nombre de vèrtexs (V) ha de ser igual al nombre d'arestes (A) més el nombre d'anells (R, total de poligons interiors a totes les cares de l'objecte, o, el que és el mateix, la diferència entre el total de polígons i el total de cares a la representació) més dues vegades el nombre de components connexes (S) menys el nombre de forats passants (H). El nombre de components connexes és el nombre de parts separades de que consta l'objecte, i habitualment val 1; dins de cada component connexa és possible anar d'un punt interior qualsevol a un altre sense sortir del component.

Aqui teniu alguns exercicis per comprovar que es cumpleix l'Equació d'Euler:

- Dibuixeu un poliedre que tingui un forat passant (H=1) i un sol anell (R=1).
- Dibuixeu un poliedre que tingui un forat passant (H=1) i cero anells (R=0).
- Dibuixeu un poliedre que tingui un forat passant (H=1) i dos anells (R=2).
- Dibuixeu un poliedre que tingui un forat passant (H=1) i que s'hagi obtingut fent una operacio booleana de resta entre un poliedre en forma de "U" i un prisma de base quadrada.
- Dibuixeu un poliedre que tingui dos forats passants (H=2) i que s'hagi obtingut fent una operacio booleana de resta entre un poliedre en forma de "U" i un prisma de base quadrada.

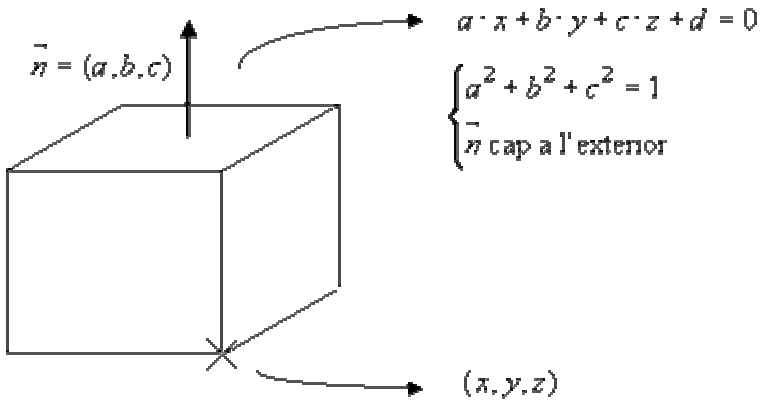
2.2 Representació per fronteres (BRep)

La representació per fronteres (BRep: Boundary Representation), és tota una família de representacions que modelitza la superfície externa de l'objecte. Enmagatzemen dos tipus d'informació: geomètrica (eqüacions dels plans, posició dels vèrtexs, etc) i topològica (veïnitat entre els elements geomètrics).

2.2.1 Geometria i topologia

<u>Geometria</u>:

La informació geomètrica que es guarda fa referència als plans de suport de les cares i a la posició dels vèrtexs:

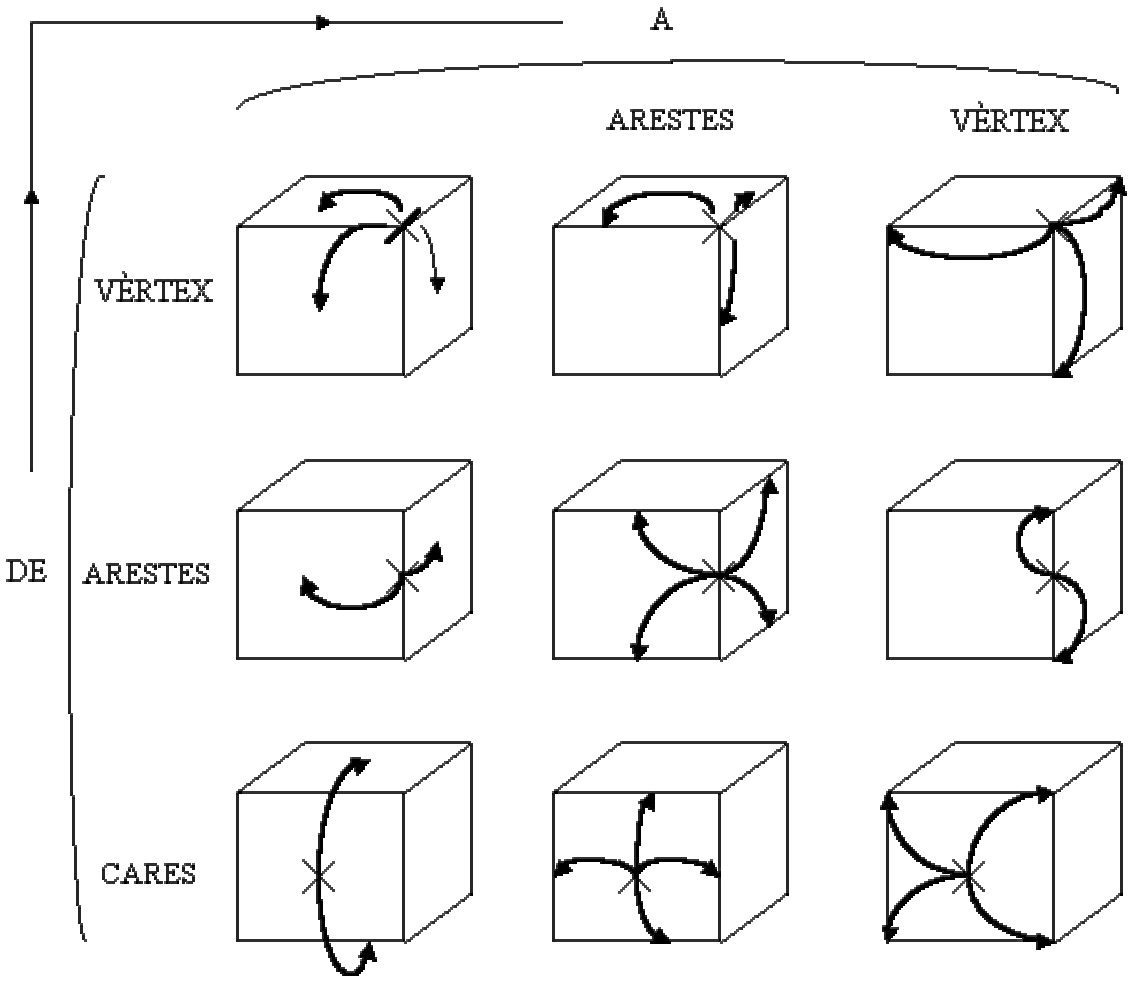


Topologia:

Les relacions topològiques les podem resumir en la següent taula:

De \ A	Cara	Aresta	Vèrtex
Cara	C: {C} 1:N	C: {A} 1:N	C: {V} 1:N
Aresta	A: {C} 1:2	A: {A} 1:N	A: {V} 1:2
Vèrtex	V: {C} 1:N	V: {A} 1:N	V: {V} 1:N

On per exemple, en la casella Aresta-Cara, tenim A: {C}, que vol dir que a partir de una aresta (A) s'ha de poder obtenir les cares ({C}) on hi és aquesta aresta. La cardinalitat indica que a partir d'una aresta obtenim dos cares. Gráficament:



Geometría i topología en BRep:

Hi ha diverses formes d'implementar el model de fronteres, però totes han de garantir que es pot obtenir la informació descrita adalt. És obvi que no cal implementar explícitament totes les relacions topològiques, sino que algunes es poden obtenir a partir d'altres. Concretament estudiarem el model jeràrquic de cares planes.

Es parteix del fet que un objecte és un conjunt de cares, unides entre elles per una aresta. Cada cara està formada per un polígon extern i eventualment, un o més polígons interns (forats). Els polígons de les cares estan representats per la successsió ordenada de les seves arestes (semi-arestes com s'explicarà més endavant). Les semi-arestes estan representades mitjançant el seu vèrtex inicial.

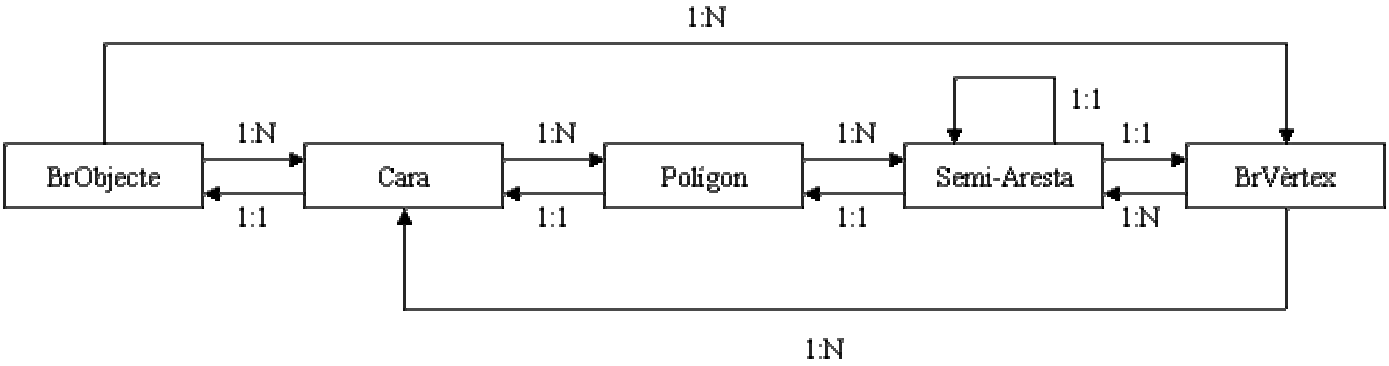
Els objectes (sòlids) tenen dins i fora i per tant podem mesurar el seu volum. Són entitats 3D

Les cares són entitats 2D. Podem mesurar la seva superficie. Poden tenir forats, que generen polígons interns. Modelen les peces de cartulina que podem enganxar per tal de generar un model físic de l'objecte.

Els polígons i les arestes són entitats 1D. Només podem mesurar la seva longitut o perímetre.

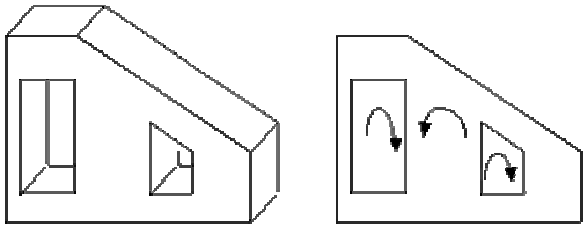
Els vèrtexs són entitats 0D. No tenen mesura

L'esquema següent il·lustra aquestes relacions topològiques:

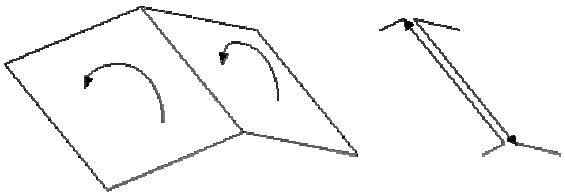


La informació geomètrica que es guarda és a nivell de cara i de vèrtex: l'equació del plà de suport i la posició a l'espai respectivament.

Les semi-arestes estan ordenades cíclicament en sentit antihorari pels polígons extern i en sentit horari pels forats:



Com la frontera és tancada, les cares que la composen són adjacents i cada aresta és compartida per dues úniques cares. La orientació d'una aresta en una cara és oposada a la seva orientació en la cara adjacent que la comparteix: es parla per això de semi-arestes per referir-se a arestes orientades.



Finalment, cada semi-aresta permet l'accés al seu vèrtex inicial segons la seva orientació.

2.2.2 Algorisme de creació d'un objecte BRep

Per crear un objecte en representació de fronteres (BRep) cal simplement omplir les corresponents llistes d'entitats geomètriques (cares, polígons, arestes i vèrtexs) i assegurar-se que es cumpleixen les tres condicions que es requereixen en els poliedres dos-varietat:

- Que tota aresta tingui dues i només dues cares veines
- Que donades dues cares, o bé tenen intersecció buida, o bé la seva intersecció (en cas que siguin veines) és l'aresta que comparteixen
- Que es cumpleix l'equació d'Euler: **C+V = A + R + 2*(S-H)**

Nosaltres suposarem que tenim dues primitives bàsiques per omplir la informació del BRep d'un objecte:

- **AfegirVertBrObj** (ob, V) , que afegeix el vèrtex V a l'objecte ob. També podem dir **AfegirVertexs** (ob, TaulaVertexs, n), que afegeix els n vèrtexs de la taula
- **AfegirCara** (ob, LlistaIndexos), que afegeix una cara formada per un cicle tancat de vèrtexs en l'ordre que apareixen a la llista. Evidentment, la llista pot ser de indexos o bé de punters. Observeu que això implica numerar els vèrtexs i saber en quin ordre els hem anat afegint.

La representació interna de l'objecte pot ser molt diversa, i en cada cas caldrà una implementació diferent d'aquestes primitives. L'únic que cal garantir és que podrem respondre a totes les interrogacions geomètriques i topològiques bàsiques a que ha de respondre un model BRep. Aqui teniu algunes d'aquestes possibles representacions:

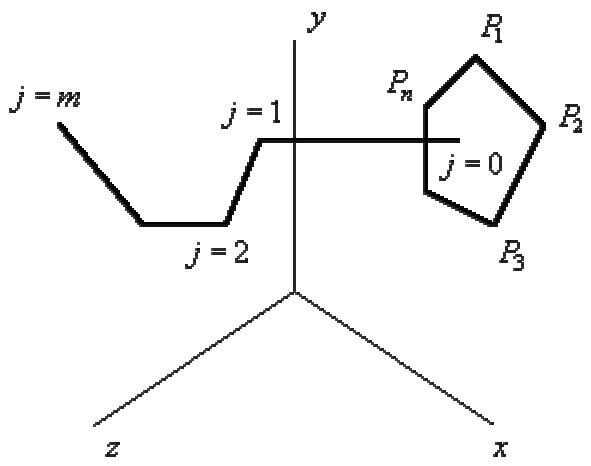
1. Una llista de cares i una llista de vèrtexs. Cada cara es representa com una llista (ordenada cíclicament) de punters o indexos a vèrtexs, i cada vèrtex es representa simplement amb les seves tres coordenades
2. Una llista de vèrtexs. En aquesta representació, que s'anomena inversa, les cares són implícites. Cada vèrtex es representa amb les seves coordenades i una llista de identificadors de les seves cares veines, ordenada cíclicament.
3. Quatre llistes: de cares, de polígons, de semiarestes i de vèrtexs. Una cara és un vector normal i una llista de polígons, un poligon és una llista de semiarestes, una semiaresta són dos punters a vèrtex i un vèrtex es representa per les seves tres coordenades.

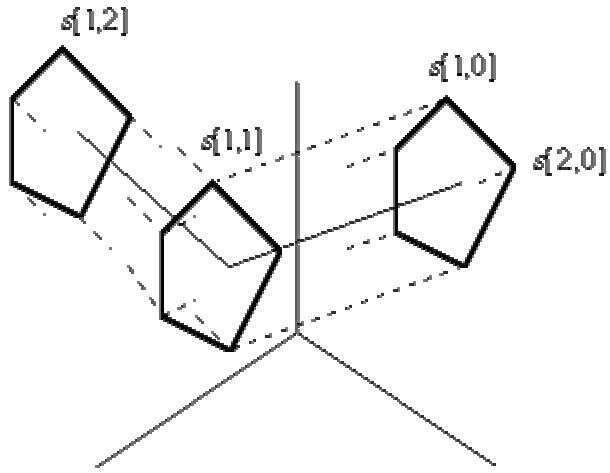
En cada un d'aquests casos, podeu comprovar (i és recomanable fer-ho com exercici) que podeu implementar algorismes específics per a respondre a totes i cada una de les següents consultes i operacions

- Equació d'una cara c
- Coordenades d'un vèrtex
- Vector normal cap enfora d'una cara qualsevol c
- Donada una cara, generar la llista de les seves cares veines, de les seves arestes veines o bé dels seus vèrtexs veins
- Donada una aresta, generar la llista de les seves cares veines, de les seves arestes veines o bé dels seus vèrtexs veins
- Donada un vèrtex, generar la llista de les seves cares veines, de les seves arestes veines o bé dels seus vèrtexs veins
- Afegir un vèrtex
- Afegir una cara

2.2.3 Algorisme genèric d'escombrat

Es tracta de generar un objecte BRep a partir de traslladar un polígon tancat al llarg d'una poligonal donada. La següent figura il·lustra millor les dades inicials:



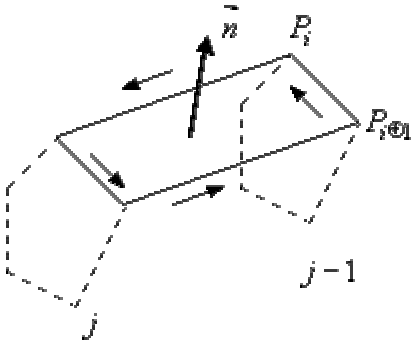


L'objecte final, *sweep(Pol)*, es defineix com:

$$Q \in sweep(Pol) \Leftrightarrow Q \text{ en } Pol(t), \text{ on } 0 \le t \le m$$

Pol(t) es calcula com una interpolació lineal entre *Pol*(int(*t*)) i *Pol*(int(*t*)+1), és a dir, l'objecte final recull tots aquells punts que en el desplaçament del polígon per l'espai (al llarg de la poligonal), en algun instant *t*, pertanyen al seu interior.

(*) Ordre en que es construeixen les cares laterals:



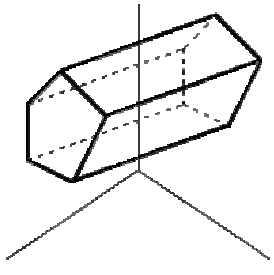
Es pot comprovar que aquest objecte té n*m + 2 cares i que compleix la fòrmula d'Euler:

$$Euler: \begin{cases} C = 2 + n \cdot m \\ V = n \cdot (m+1) \\ A = (m+1) \cdot n + n \cdot m \\ R = 0, S = 1, F = 0 \end{cases}$$

S'ha de destacar que els nous punts de l'objecte s'obtenen mitjançant una transformació geomètrica dels punts inicials. El tipus d'escombrat que volguem fer ens determinarà el tipus de transformació que apliquem als punts:

Translació:

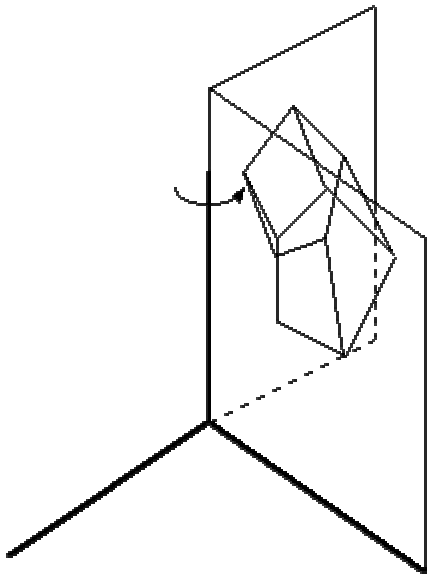
Simplement una translació de l'objecte entre dos punts (una poligonal formada per una recta). En aquest cas m=1, la transformació geomètrica és una translació i no hi ha possibilitat d'interseccions:



Rotació:

L'objecte que volem obtenir s'obtindrà rotant els punts del polígon inicial al voltant d'un eix:

En aquest cas la transformació geomètrica és una rotació donat un angle (angle α*j). En principi es donarà un angle inicial θ_i tindrem que $\alpha = \theta / m$. Es pot augmentar la qualitat incrementat 'm'. Com és obvi, cal que $\theta = \alpha j \leq 2\pi$, i per tant sorgeix un cas especial: quan l'objecte és tancat. En aquesta situació no s'han d'afegir les cares inicial i final.



Altres punts a destacar són:

- No hi ha possibilitat d'intersecció entre cares
- Quan s'afegeixen les cares laterals, ara la "j" passa a ser cíclica, i per tant s'ha d'anar en compte fent j-1.

Dirigit per poligonal:

És una barreja dels dos anteriors:

- S'ha de fer una rotació per colocar el polígon paral·lel al plà del següent punt de la poligonal
- S'ha de fer una translació per moure el polígon cap al següent punt de la polígonal

Encara queda un grau de llibertat del polígon dins del plà per minimitzar la dorsió del poliedre.

2.2.4 Algorismes de subdivisió recursiva (Doo i Sabin)

Aquest algorisme parteix d'un poliedre **P** (que suposarem que no té anells i que per tant R=0. A més, suposarem que té una sola component connexa, S=1) i genera un nou objecte (poliedre) de sortida, treballant en tres fases:

- Per cada cara **c** del poliedre **P**, generem una nova cara encongida respecte la cara original i l'afegim al nou objecte. Si la cara **c** tenia les semiarestes a_1 .. a_n i els vèrtexs v_1 .. v_n, direm que la cara encongida té les semiarestes a'_1 .. a'_n i els vèrtexs v'_1.. v'_n, on a'_k = Encongit(a_k) i v'_k = Encongit(v_k).
- Per cada aresta **a** del poliedre **P** corresponent a dues semiarestes **ad**, **ae**, generem una nova cara de 4 arestes que uneix les dues semiarestes Encongit(**ad**) i Encongit(**ae**), i que s'han obtingut després d'haver encongit, a la fase anterior, les dues cares contigües a l'aresta **a**.
- Per cada vertex **v** del poliedre **P**, generem una nova cara, que uneix tots els vertexs resultants de l'encongiment de v: Encongit(**v**) per totes les cares de **P** que eren veines al vèrtex **v**.

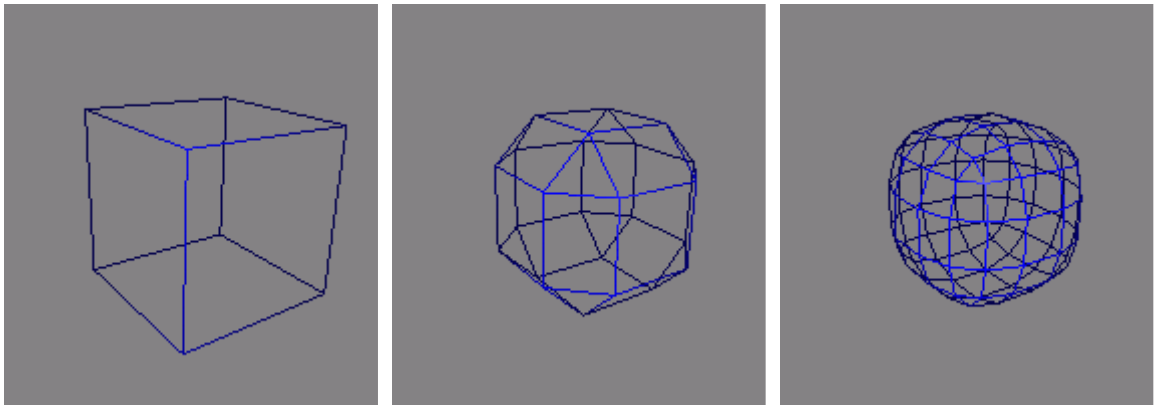
Observeu que si el poliedre **P** tenía C cares, A arestes i V vèrtexs, el nou poliedre tindrà C+A+V cares i un total de 2*A vèrtexs. El nombre de cares és obvi; pel que fa al nombre de vèrtexs, cal observar que són el de les cares encongides i que el seu nombre és el de semiarestes de **P**, ja que en el procés d'encongiment de cares veines, les semiarestes es desdoblen. Si **P** no té forats passants l'equació d'Euler és C+V=A+2, i el nombre d'arestes del nou poliedre és C+3*A+V-2 . Si les cares de **P** són convexes, les cares del nou poliedre també ho són.

Un cop hem acabat aquests tres passos, tenim un nou poliedre amb R=0, subdividit respecte l'inicial. Podem repetir el proces de subdivisió amb aquest nou poliedre, per anar-lo suavitzant.

L' únic que resta és saber com encongir una cara. Doo i Sabin demostren que, per tal de maximitzar la suavitat del resultat (de fet, demostren que en les zones en que els vèrtexs tenen quatre cares veines, la superficie tendeix a un B-Spline biquadràtic), en una cara amb n vèrtexs v_1.. v_n,

$$v'_k = (\text{Suma} (\text{Alfa}_{k-i} * v_i)) / n$$

On la suma s'estén a tots els n vèrtexs de la cara. Els coeficients Alfa són tots positius i sumen la unitat, de manera que els vèrtexs encongits són una mena de ponderació dels vèrtexs inicials de la cara. El coeficient Alfa_0 és igual a $\text{Alfa}_0 = (1/4 + 5/(4*n))$ i els altres es calculen com $\text{Alfa}_{k-i} = (3 + 2 * \cos (2*\pi*(k-i)/n)) / (4*n)$.

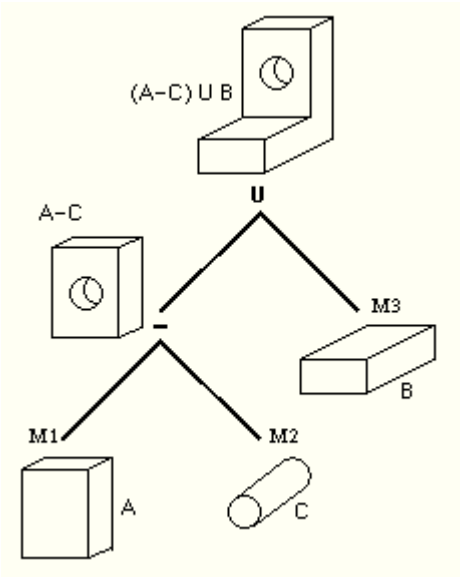


Exemple Doo-Sabin: Cub

2.3 Geometría constructiva de sòlids: Arbres CSG

És un model de sòlids basat en una estructura en arbre amb:

- Nodes interiors: Operacions Booleanes (Unió, Intersecció, Diferència)
- Nodes terminals (fulles): objectes primitius parametritzats (cub, con, esfera, cilindre, etc)



Exemple:

De vegades també s'inclou una transformació lineal (matriu) associada als nodes o només a les fulles. En cas contrari, les primitives han de ser parametritzades (radi, alçada, posició, ...) Habitualment l'arbre és binari, però per les operacions d'unió i intersecció no caldria.

Exemple d'expressions algebraiques que modelitzen l'objecte:

A := Bloc(a1,b1,c1); A := TG(A,M1)
B := Bloc(a2,b2,c2); B := TG(B,M2)
C := Cilindre(r,h); C := TG(C,M3)
Obj := (A-C) U B

2.3.1 Característiques del model:

- **No ambigu** i **vàlid**, sempre que les operacions booleanes siguin les regularitzades.
- **No únic**. Per exemple, en la figura es podria haver canviat l'ordre de la unió i la diferència ((AUB)-C) o haver obtingut el perfil de la "L"com la diferència entre un bloc gran i un de més petit.
- Molt **concís** (objectes complexos es poden representar utilitzant poca memòria).
- **Domini de representació**: depèn de les primitives admeses. En general, els objectes representables estaran sempre limitats per (fragments de) superfícies de les primitives. Admet objectes que no siguins 2-varietat (*non-manifold*).

2.3.2 Operacions i interrogacions sobre el model:

- **Transformacions geomètriques**: immediates, modificant les matrius o els paràmetres. Fins i tot de part d´un objecte.
- **Classificació punt-sòlid:** implica saber classificar un punt respecte a les primitives. Vegeu l'algorisme.
- **Classificació recta-sòlid:** ray-casting amb intèrvals de recta. Vegeu l'algorisme.
- **Càlculs de volums, etc:** complicats. Pot caldre **avaluar** el model. Es pot fer un càlcul aproximat mitjançant *ray-casting* i llençant molts raigs.
- **Operacions booleanes:** immediates!.
- **Edició (construcció):** relativament senzilla i fàcil de manegar (el model és bastant intuïtiu, excepte potser l'operació d'intersecció).
- **Visualització:** Existeixen algorismes directes per *ray-casting* (algorisme de Roth), però si es volen utilitzar les eines estàndard (z-buffer), cal avaluar el model.

Avaluar el model significa calcular exactament quines cares, arestes i vèrtexs té, és a dir, transformar-lo en B-Rep.

2.4 Models de descomposició espaial: Arbres octals (octrees)

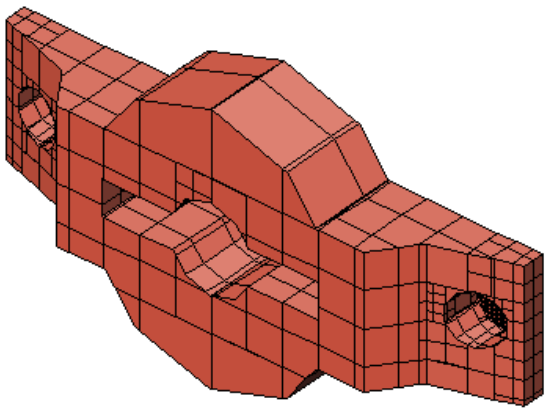
És un model de sòlids del tipus enumeració espacial. Cal partir d'un cub "univers" que conté tot l'objecte, i anar dividint l'espai en octants recursivament. El resultat és un arbre octal amb nodes interiors o exteriors en funció de si contenen una part prou "simple" de l'objecte.

- Nodes interiors: Grisos.
- Nodes terminals (fulles): Blancs, Negres, Cara, Aresta, Vèrtex, Grisos Terminals.

Un octtree clàssic només té nodes G,N,B (els nodes grisos G són nodes corresponents a cubs que cal subdividir mentre que els nodes blancs B són nodes totalment exteriors a l'objecte i els nodes negres N són totalment interiors a l'objecte).

Els octtrees extesos d'altres de més complicats. Habitualment es necessita un límit que determina el nivell màxim de divisió (el dels grisos terminals) i la precisió amb què quedaran representats els objectes. L'objecte es representa com un arbre que codifica el procés de subdivisió de l'espai, de forma que cada node de l'arbre es correspon univocament amb un cub de la subdivisió espaial. A partir de la posició d'un node a l'arbre podem inferir el tamany del cub associat i la seva posició a l'espai.

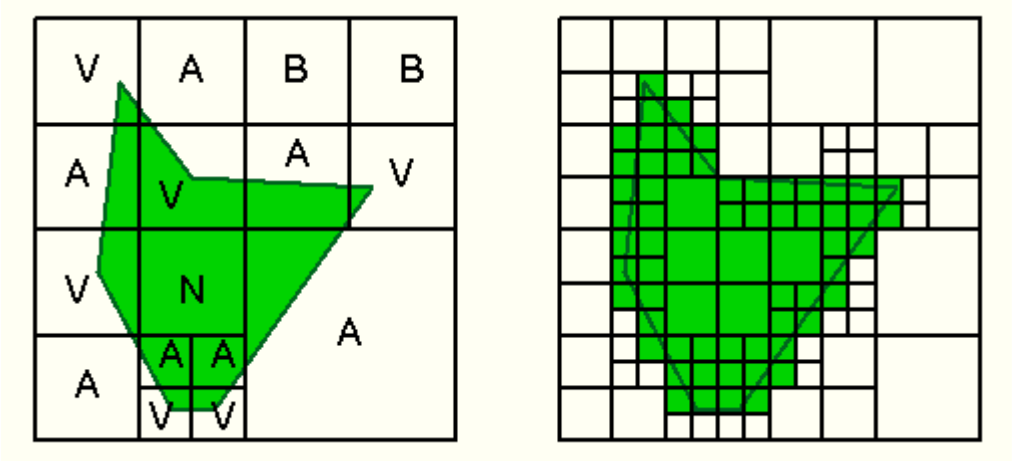
Exemple d'octree extès:



Partim del cub univers

El subdividim en 8 nodes (octants) recursivament fins que dins del node hi ha una part prou senzilla de l'objecte original.

El mateix concepte és vàlid per a dues dimensions, quadtrees, que modelitzen polígons:



Quadtree Extès **Quadtree Clàssic**

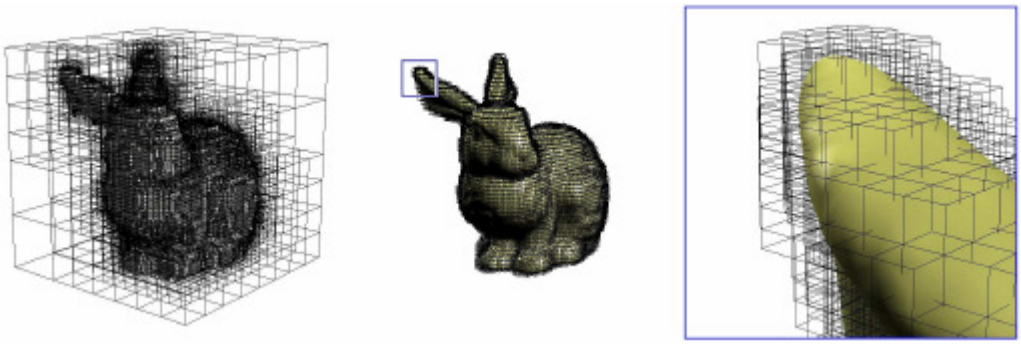
2.4.1 Característiques del model:

- **No ambigu** (fixats uns eixos i un cub univers)
- Hi ha octrees extesos als quals **no** correspon cap objecte **vàlid.**
- **Unicitat**, fixats uns eixos de referencia i un cub univers. Cal mantenir l'octtree compactat (un node gris no pot tenir tots els fills blancs, per exemple).
- **Domini de representació**: Permet representar qualsevol objecte de forma aproximada. En el cas dels octtrees extesos, el domini són els poliedres. També permet modelitzar l'interior dels sòlids (serveix com a model de volum).
- **Considerable espai de memòria**, depenent del tipus de nodes terminals que s'admetin, però pot ser bo per a representar objectes de forma molt complexa. En tot cas, si es considera com a model de volum, segur que ocupa menys espai que una voxelització.

2.4.2 Operacions i interrogacions sobre el model:

- **Transformacions geomètriques**: molt costoses!
- **Classificació punt-sòlid:** senzilla i molt ràpida (aprofitant la divisió espacial).
- **Classificació recta-sòlid:** tampoc gens costosa i ràpida.
- **Càlculs de volums, etc:** senzills, però aproximats en els octrees clàssics. Permeten fer aproximacions ràpidament.
- **Operacions booleanes:** bastant senzilles i ràpides (aprofitant la divisió espacial); tant sols cal saber operar dos tipus de nodes fulla qualssevol.
- **Edició (construcció):** algorísmicament, partint d'una representació en algun altre model.
- **Visualització:** es pot utilitzar un mètode ad hoc que aprofiti l'ordenació espacial que proporciona (algorisme del pintor) o fer ray-casting aprofitant la rapidesa del test d'intersecció amb una recta, però visualitzar-los aprofitant un z-buffer és senzill perquè només cal saber visualitzar els diferents tipus de node fulla. Admeten la gran majoria de tècniques de visualització pròpies dels models de volum (mètodes projectius i mètodes de traçat de raig) i n'acceleren alguns d'ells.

Habitualment s'utilitza com a model auxiliar per aprofitar la localització espacial. Per exemple, per accelerar les operacions booleanes entre B-Reps. En aquest cas, cada node terminal podria contenir informació de les cares i arestes contingudes dins del seu cub associat, per tal de poder accelerar els tests cara-aresta.



Nodes terminals *Nodes grisos terminals* *Ampliació*

2.5 Models de volum

Els models de volum són útils quan volem modelar camps escalars o vectorials. Un camp és una magnitut **w** que té un valor a cada punt de l'espai,

$$\mathbf{w = f(x,y,z)}$$

Observeu que la informació que volem guardar és molt més gran que la que tractavem fins ara. No volem modelar la superficie dels objectes, sino el valor d'una o més magnituts a cada un dels punts del seu interior!

Exemples de camps, que caldrà modelar com models de volum:

- Qualvevol magnitut meteorològica: pressió, densitat de l'aire, temperatura, humitat, etc.
- Grau de contaminació en una certa zona de l'atmòsfera
- En geología, la concentració d'un cert mineral a cada punt del subsòl
- En medicina, la densitat dels teixits a cada punt de l'interior d'un òrgan, o bé el seu grau d'irrigació sanguínea, etc

2.5.1 Voxels

Són la manera més senzilla i estesa de modelar volums. Se suposa que l'espai a modelar es cúbic (o paralelepipèdic) i es divideix en una malla regular de N x N x N. El camp w=f(x,y,z) es modela mitjançant una matriu W[0..N, 0..N, 0..N] que emmagatzema els valors del camp w a cada un dels vèrtexs de la malla. Aixi, el valor W[i,j,k] representa el valor del camp w al vèrtex de la malla de coordenades x[i], y[j], z[k].

Observeu que estem discretitzant una magnitut 3d de la mateixa manera que, en una imatge digital, discretitzem una fotografía 2d. Un món de vòxels és la generalització 3d d'una imatge digital. Aixi com "pixel" ve de "picture element", vòxel ve de "volume element".

En cas que tinguem regions amb valors del camp molt uniformes, pot tenir sentit compactar el model (món) de vòxels en un arbre octal (octree, vegeu l'apartat 2.4). Si tenim vuit vòxels veins que formen un cub de tamany doble i el seu valor (dins d'una tolerància donada) és el mateix, podem compactar-los i treballar directament amb el seu node pare a l'octree. Aquest procés de compactació es pot anar repetint fins generar tot l'arbre.

2.5.2 Models de vòxels i octrees d'objectes sòlids

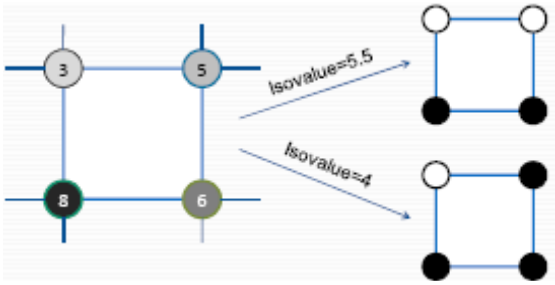
Donat un sòlid S, podem calcular el seu model de vòxels. El que passa és que ara w només podrà tenir dos valors, dins i fora. Aquest model de vòxels també es pot compactar per generar un octree. Aquest octree pot ser útil per accelerar càlculs geomètrics: punt dins de sòlid, operacions booleanes, anàlisi de interferències i de colisions. En tot cas, qualsevol esquema de subdivisió de l'espai pot ser útil si el que es vol és l'acceleració de càlculs geomètrics. A banda dels arbres octals, també s'utlilitzen els arbres BSP i els Kd-trees.

2.5.3 Algorisme Marching Cubes

Marching Cubes (MC) es un dels algorismes mes utilitzats per extreure una superficie a partir d'un model de voxels.

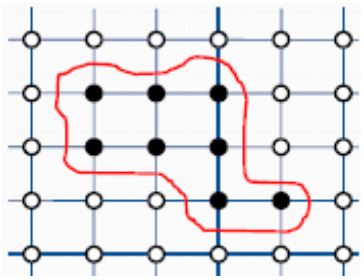
Dades de partida:

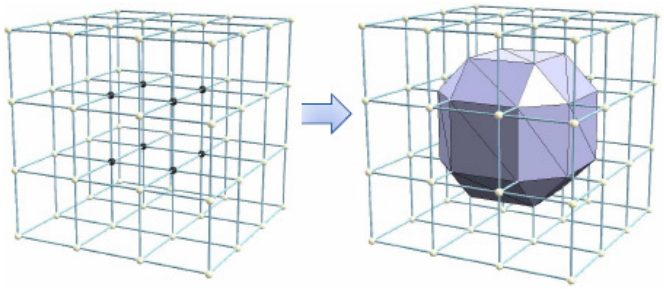
- L'entrada de l'algorisme és un model de voxels que representa un camp escalar v=f(x,y,z)
- El camp escalar de partida pot ser o no binari. Si el model de partida no es binari, cal proporcionar un parametre addicional (umbral o isovalue) que permet classificar els punts en interiors i exteriors



Superfície generada:

- Volem generar una superficie que separi els punts interiors dels exteriors (model binari)
- La superficie que generem a partir d'un model de volum, sempre:
  - Ha de separar els punts interiors dels exteriors
  - Per tant ha de ser orientable i tancada



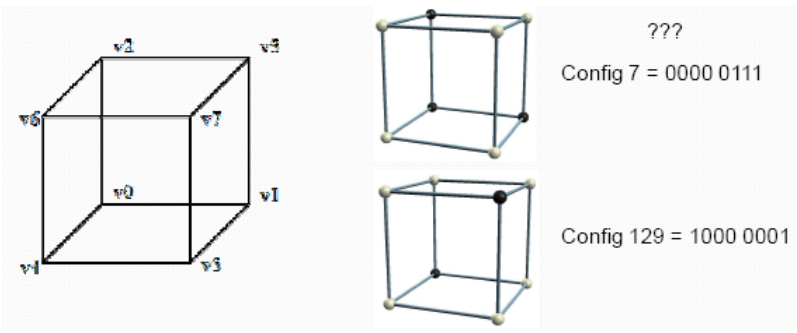


Idea basica:

- Anar tractant tots els cubs formats per 2x2x2 mostres veines de la graella
- Per cada cub es reconstrueix la part de la isosuperficie interior al cub (tots els triangles de la isosuperficie que genera MC pertanyen a un unic cub)

Configuracions i casos:

- Casa cub té 8 vèrtexs blanc/negre → 28 = 256 configs
- Si numerems els vèrtexs, cada config es pot representar en 1 byte
- Moltes configuracions son simetriques i no cal considerarles per separat
- Casos article MC original: 14+1

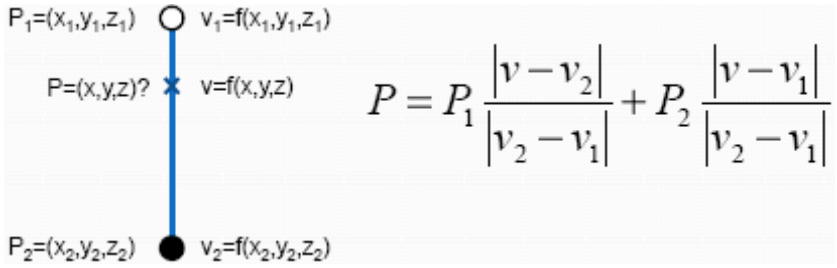


Algorisme de Marching Cubes:

Per cada cub que processa l'algorisme, cal reconstruir…
- Geometria: vèrtexs de la isosuperficie
- Topologia: triangles que connecten els vèrtexs

1. Creació dels vèrtexs

Marching Cubes crea un vertex de la isosuperficie per cada aresta de la graella amb un canvi de signe. La posició exacta del vèrtex dins l'aresta es determina mitjançant interpolació lineal:



$$P = P_1 \frac{|v - v_2|}{|v_2 - v_1|} + P_2 \frac{|v - v_1|}{|v_2 - v_1|}$$

2. Creació dels triangles

Un cop hem creat tots les vèrtexs, cal saber com els hem d'unir per formar els triangles interiors a cada cub. Alguns casos son trivials… d'altres no, n'hi ha d'ambigus:

- Els que tenen una cara ambigua (dos vèrtex blancs i dos negres, en diagonal ≈ totes les arestes amb canvi de signe)

- Els que tenen dos vertexs blanc (o negre) en qualsevol de les diagonals del cub

Marching Cubes utilitza una LUT (look-up table) de 256 entrades (una per cada configuracio), que indica com cal reconstruir els triangles dins del cub:

- Número de triangles
- Per cada triangle: Indexs (a,b,c) dels vèrtexs del triangle. Cada index es un valor 0..11 que indica la aresta del cub sobre la qual esta el vèrtex.

3. Construccio de la LUT

- Es fa a partir d'una anàlisi dels 14+1 casos
- Per cada cas ambigu, els autors van triar la reconstruccio més senzilla
- Cada cas → entre 1 i 4 triangles

```
per cada mostra (i,j,k)
       config := determinar configuracio del cub (i,j,k) → (i+1, j+1, k+1)
       reconstruccio := LUT[config] // accedim a la taula precalculada
       // reconstruccio te per exemple la forma {{a,b,c}, {d,e,f}...}
       crear els vertexs amb interpolacio lineal
       // hem creat un vertex per cada aresta (a, b, c, d...)
       crear els triangles indicats a la LUT
       // haurem creat els triangles corresponents (a,b,c), (d,e,f)
       // haurem de substituir els indexs a aresta
       // (entre 0 i 11) pels indexs als vertexs creats
fper
```

Interpolació de MC:

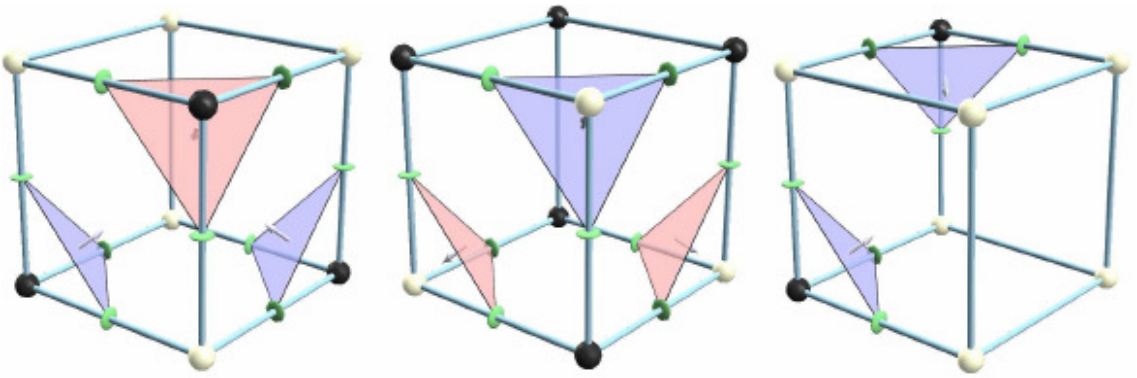
- Interpolació lineal per fixar la posició dels vèrtexs dins les arestes.
- Una superfície lineal (triangles) per unir aquests vèrtexs.

Problemes de MC original:

- Numero de triangles de la superficie generada
- Qualitata dels triangles → Pot crear arestes de longitud 0 i triangles d'àrea nula
- Hi ha un cas repetit (11=14)
- La superfície generada pot tenir forats! La superfície pot no ser tancada! Es degut a que algunes cares ambigues no es reconstrueixen de forma consistent

- <u>Reconstruccio de les cares ambigues</u>:

  En alguns casos els vèrtexs negres queden separats (ex. cas 3)
  En d'altres casos els vèrtexs negres queden junts (ex. cas ~7)

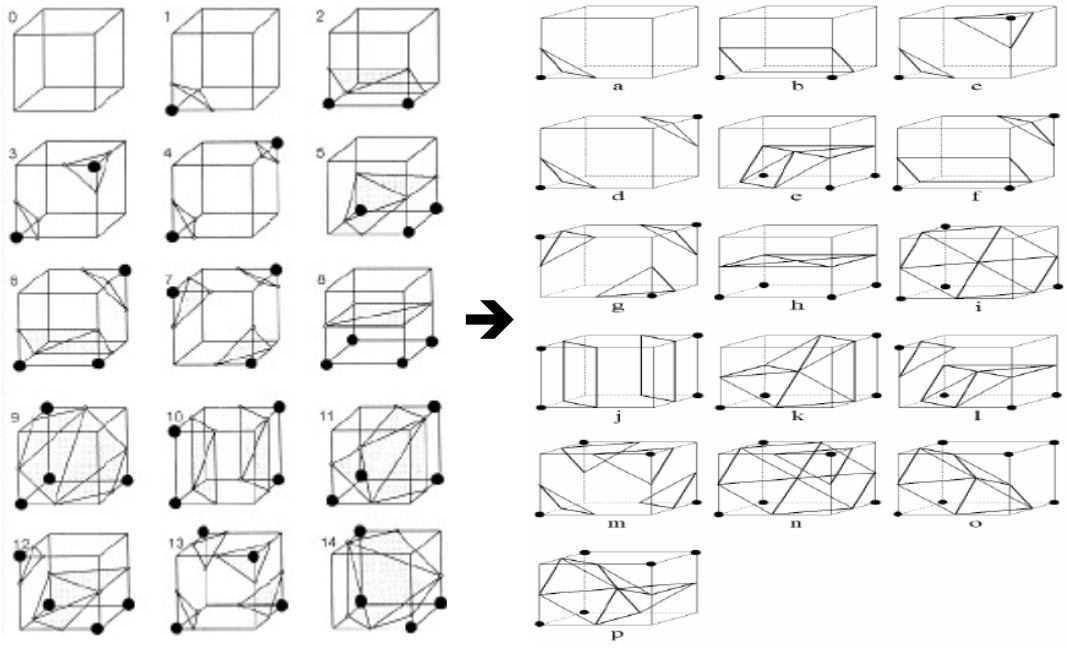


Cas 7 *Cas 7 complementat* *Cas 3*

Solució:

- <u>Taula de casos modificada</u>:

  Cal garantir que, per qualsevol parella de configuracions, la reconstrucció de la cara comú és consistent.



3 MALLES DE TRIANGLES

3.1 Representació i propietats

Una malla de triangles es un conjunt connex de triangles. Si el nombre de triangles és prou gran, les malles poden representar amb suficient aproximació objectes curvats o cilindrics, superficies de Bézier, NURBs, etc.

3.1.1 Estructura del model

El model més senzill d'un objecte representat per una malla de triangles és el format per dues llistes, una de triangles i una de vèrtexs:

- Cada triangle consta de tres punters a vèrtexs (a més d'altres informacions com color..)
- Cada vèrtex consta de les seves tres coordenades (x,y,z)

Les malles poden ser obertes o tancades. A les malles tancades, cada triangle té tres triangles adjacents a través de cada una de les seves arestes. En canvi, les malles obertes tenen vores, i les arestes de les vores delimiten només amb un triangle. Les malles tancades, si els triangles no s'intersecten, tanquen un volum i són fronteres de sòlids. En aquest cas, com que el nombre de semiarestes és tres cops el nombre de cares, es cumpleix (C és el nombre de cares, A és el nombre d'arestes i V el nombre de vèrtexs de la malla):

$$2\,A = 3\,C$$

D'altra banda, com que R=0, si suposem que S=1 i F=0, l'equació d'Euler ens diu que,

$$C + V = A + 2$$

Agrupant les dues equacions, tenim una relació entre el nombre de vèrtexs i el nombre de cares a tota malla tancada:

$$2\,V = C + 4$$

3.1.2 Generació de malles

Les malles triangulars es poden generar:

- Per subdivisió a partir de models de sòlids amb cares planes (poliedres). Els algorismes més coneguts són el de Doo & Sabin i el de Loop.
- Per generació d'isosuperficies a partir d'un model de voxels.
- Per facetització de superficies. Tot seguit donem un algorisme de facetització d'una superficie on $umin \leq u \leq umax$ i $vmin \leq v \leq vmax$. Existeixen algorismes más sofisticats, basats en una triangulació adaptativa i de Delaunay.

3.1.3 Simplificació

Existeixen molts algorismes que redueixen el nombre de triangles d'una malla, obtenint una altra malla més simple i propera a la inicial. En alguns d'aquests algorismes es pot especificar una tolerància, i l'algorisme ens assegura que la malla simplificada es mantindrà a una distància sempre menor que la tolerància respecte a la malla inicial. Veure també l'apartat específic del temari sobre simplificació (apartat 3.4).

3.1.4 Tires de triangles (strips)

Si una malla de triangles es representa com una tira de triangles contigus, la informació que cal (un cop s'ha codificat el primer triangle de la tira) és simplement un vèrtex i un bit per triangle. Aquesta és una representació molt compacta, admesa per exemple per OpenGL. Existeixen algorismes eficients per a la conversió d'una malla en tires de triangles.

3.1.5 Compressió i transmissió progressiva

Existeixen algorismes molt eficients per a la compressió de malles, que optimitzen el temps de transmissió a través de la xarxa (per exemple).

3.1.6 Relació amb els models de punts i amb els models de volum

Tal com hem vist, una **malla de triangles** es representa habitualment amb dues llistes (o taules): una de triangles i una de vèrtexs. Cada triangle consta de tres punters a vèrtexs (a més d'altres informacions com color..), i cada vèrtex ve representat per les seves tres coordenades (x,y,z).

Un model de **núvol de punts** és simplement una taula o llista de punts. Cada punt pot tenir informació addicional a part de les seves coordenades, com pot ser el color, el seu vector normal, etc. La diferència bàsica amb una malla de triangles és que aqui no tenim informació sobre la connectivitat entre els punts.

Un **model de volum (voxelització)** és una taula que conté informació sobre una malla regular de N x N x N cel.les uniformes que divideixen un cub univers a l'espai. Aquesta malla està formada per cel.les, totes elles del mateix tamany, que estan limitades per arestes i vèrtexs de la malla. Hi ha moltes variants de models de volum (codificació d'un o més objectes a través de la informació dins_objecte_k / fora als vèrtexs de la malla, codificació d'un o més objectes a través de la informació dins_objecte_k / fora als vèrtexs de la malla i de la informació dels punts de tall i vectors normals entre la superficie dels objectes i les arestes de la malla, etc.). Aquí ens concretarem als models de volum que representen camps escalars i que guarden un valor escalar a cada un dels vèrtexs de la malla: **taula [0..N, 0..N, 0..N] de float.** Un cas particular dels models de volum que representen camps escalars són els camps de distàncies. En aquest cas, el valor que es guarda a cada vèrtex de la malla es el valor de la distància (amb signe o sense signe) a la superficie de l'objecte que estem representant. En canvi, direm que una voxelització és binaria si la informació a cada vèrtex és del tipus dins/fora.

Podem convertir una malla de triangles en un núvol de punts simplement eliminant la llista de triangles. En el cas de triangles massa grans, pot ser bó fer un mostreig més fí i generar punts dins de cada triangle.

Convertir en canvi un model de punts a una malla de triangles no és tant trivial. Es l'anomenat problema de la reconstrucció. Hi ha diverses solucions, com els algorismes dels Alpha Shapes (i derivats), els algorismes basats en la reducció d'una membrana discreta o els algorismes de Hornung i Kobbelt, basats el el càlcul del tall minimal del camp de distàncies definit per una crosta discreta. Els núvols de punts es poden operar i visualitzar, vegeu per exemple el projecte PointShop: www.pointshop.com

El càlcul d'un model de volum (camp de distàncies) a partir d'una malla de triangles tampoc és complexe. El que cal és calcular la distància mínima desde cada punt de la malla als triangles de la malla. El càlcul es pot accelerar si es calcula primer la distància a la malla desde els vèrtexs de les cel.les de la malla que contenen triangles de la superficie, i després s'extenen aquests valors als altres vèrtexs de la malla.

Hi ha dos algorismes clàssics per passar d'un model de volum a una malla de triangles: l'algorisme de Marching Cubes i l'algorisme de Dual Contouring. Teniu més informació sobre aquests dos algorismes a l'apartat 2.5 del temari.

Els models de volum moltes vegades tenen interés com a model intermig per a certes operacions:

- Reconstrucció de superficies a partir de núvols de punts
- Reparació de models de malles de triangles
- Simplificació de models de malles de triangles (apartat 3.4 del temari)

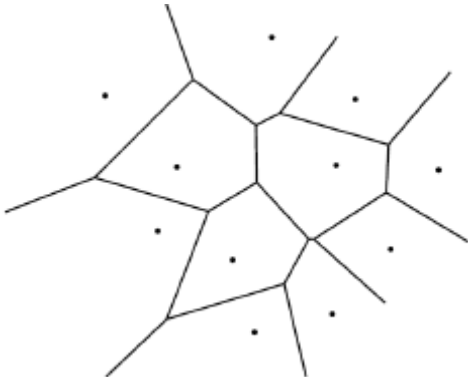
3.2 Algorismes de triangulació

3.2.1 Diagrames de Voronoi i triangulacions de Delaunay

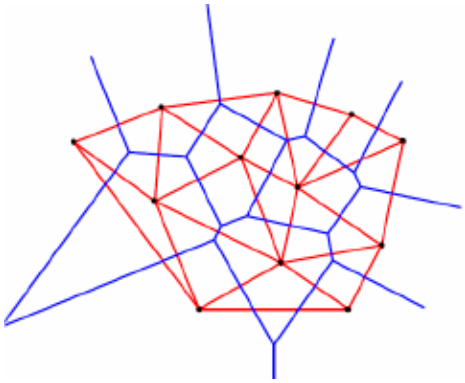
Donat un conjunt de punts en un pla (que poden ser els vèrtexs d'un poligon que volem triangularitzar), el diagrama de Voronoi es un conjunt de regions R_1 ... R_n on cada regió correspon a un dels punts inicials V_1 ... V_n , tals que la regió Rk conté el conjunt de punts del pla que estan més a prop de V_k que de qualsevol altre dels punts incials donats.

Un cop construit el diagrama de Voronoi, la seva dual és la triangulació de Delaunay del conjunt de punts donats. Aquesta dualitat s'ha d'entendre com que dos punts V_i i V_j estan conectats per una aresta de la triangulació de Delaunay si i solament si les dues regions corresponents R_i, R_j són contigües i comparteixen una aresta del diagrama de Voronoi.

Els diagrames de Voronoi es poden calcular amb algorismes de "divide and conquer" o be amb l'algorisme d'en Steven Fortune, que el calcula de forma incremental a base d'una recta vertical que escombra el model d'esquerra a dreta; el diagrama de Voronoi es va construint a la part ja escombrada per la recta.



(a) *Diagrama de Voronoi*



(b) *Graf de Delaunay*

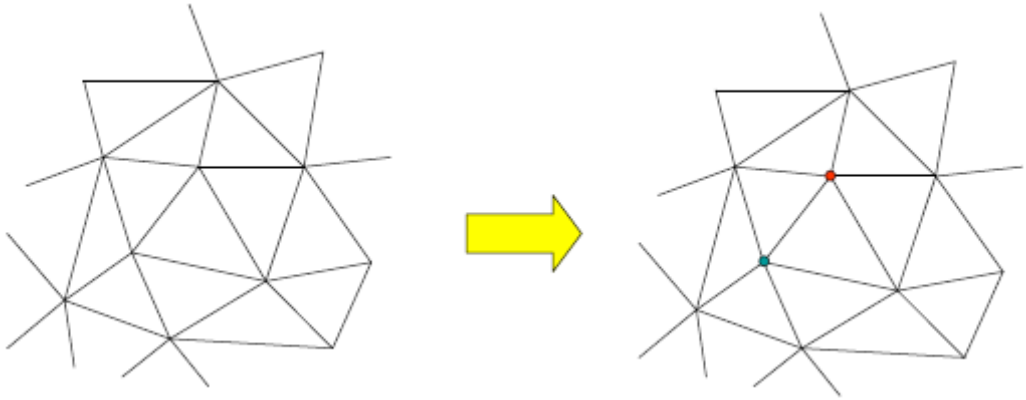
3.2.2 Triangulacions restringides de Delaunay

Una triangulació restringida de Dealunay és una triangulació tal que manté algunes arestes que s'han especificat inicialment. En el cas de la triangulació de polígons nosaltres voldrem triangular el conjunt de vèrtexs d'un polígon tot mantenint les arestes inicials del polígon.

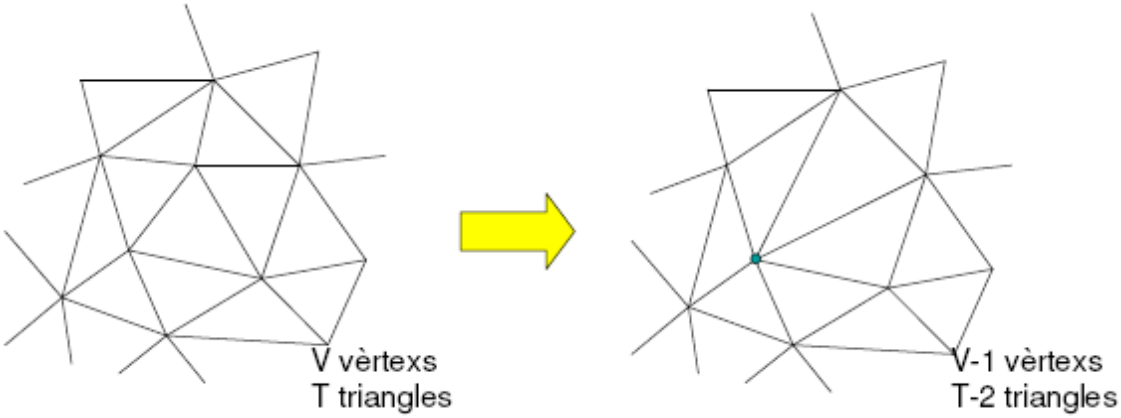
3.3 Simplificació de malles

3.3.1 Vertex Removal (utilitza malles de Hoppe)

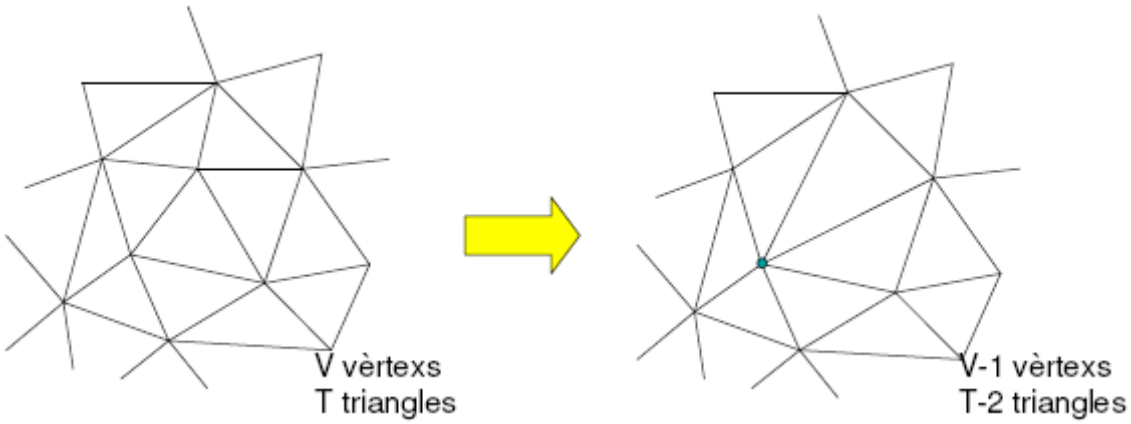
Pas 1: Escollim el vèrtex a eliminar (el vermell) I el vèrtex que el substituirà (verd).



Pas 2: S'elimina l'aresta entre els dos. Totes les arestes que anaven al vermell, van al verd.

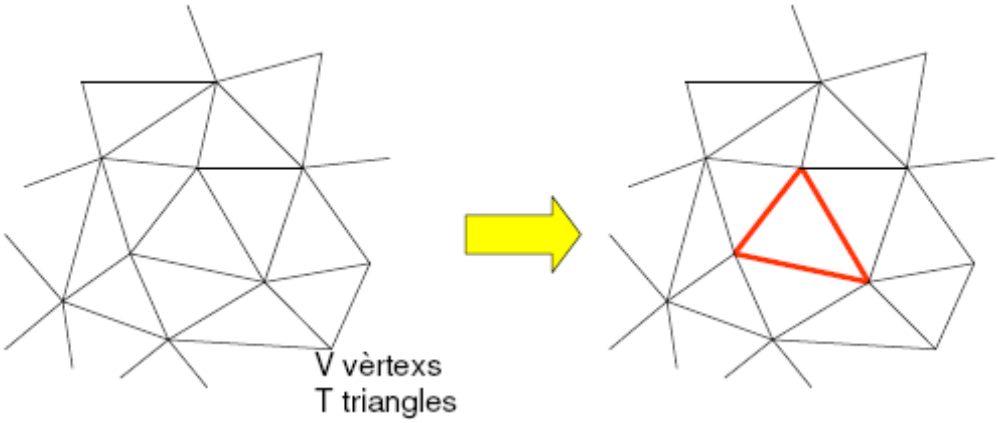


Pas N: Es pot anar simplificant fins arribar a la base Mesh.
Si a cada pas es guarda informació (index del V a expandir, coordenades del nou vèrtex, index circular de les dues arestes al voltant de V que caldrà assignar al nou vèrtex) es pot anar regenerant la malla de forma progressiva a partir de la Base Mesh.

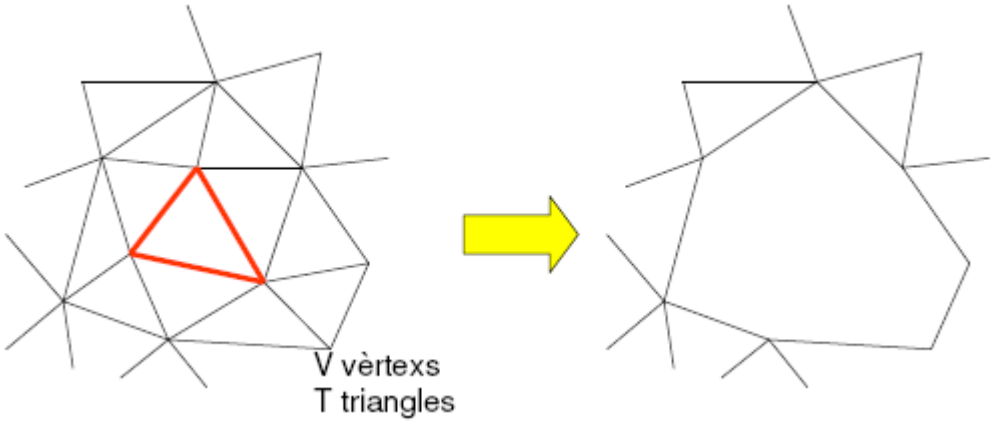


3.3.2 Face Removal

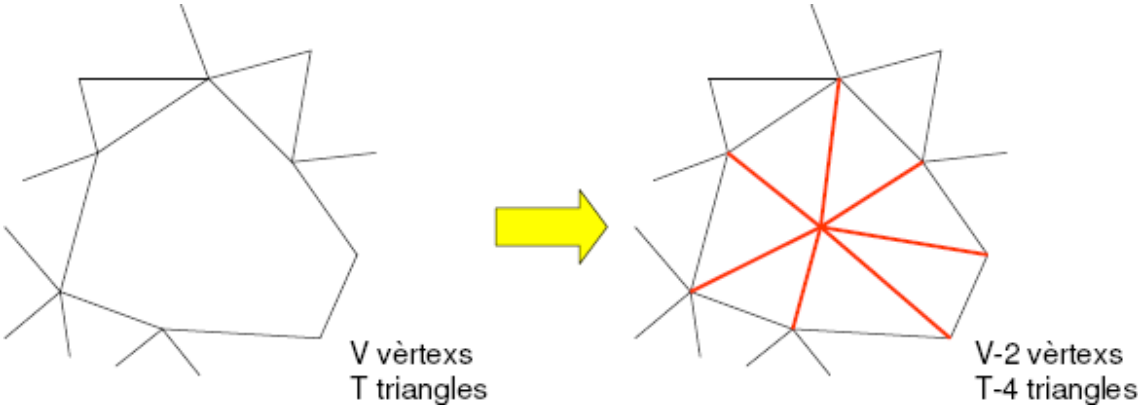
Pas 1: S'escull una cara.



Pas 2: S'esborra la cara junt amb totes les cares que comparteixen un vèrtex amb ella.

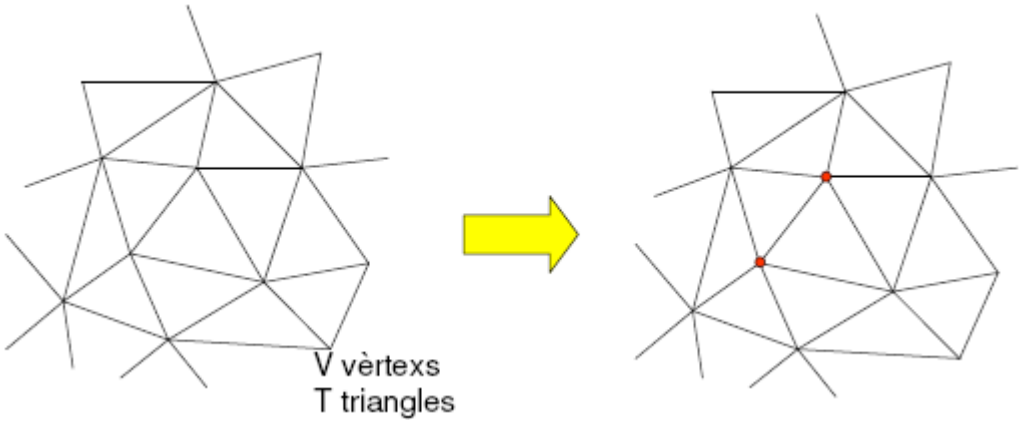


Pas 3: Es triangula el forat, després d'insertar un vèrtex "al mig" del forat.

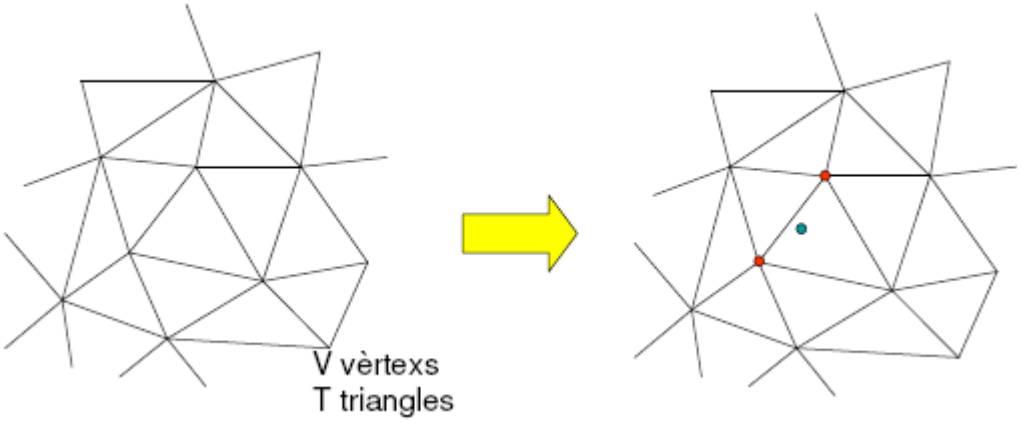


3.3.3 Colapse d'aresta (edge collapse)

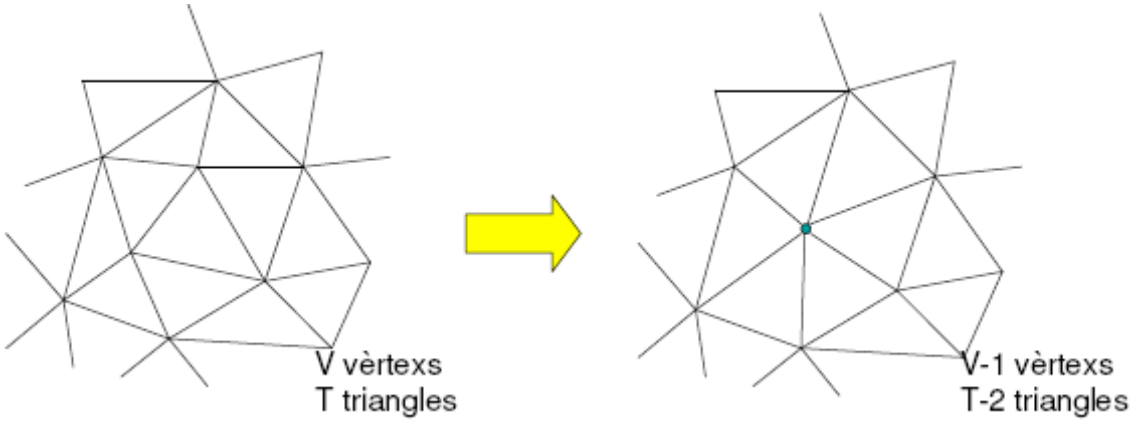
Pas 1: Escollim els dos vèrtexs a eliminar (vermells), extrems de l'aresta que decidim colapsar.



Pas 2: Decidim la posició del vèrtex (verd) que els substituirà.



Pas 3: Eliminem l'aresta que uneix els dos vèrtexs vermells.
Les arestes que confluíen a ells, passen a dirigir-se al nou vèrtex verd.



Problemes a resoldre:

- Com escollim l'aresta a eliminar
- Com calculem la posició del nou vèrtex

Solució: La Mètrica Quadràtica d'Error

- Es calcula una quàdrica per a cada triangle de la malla.
- Per a cada vèrtex, es calcula la seva quàdrica com la suma de totes les quàdriques dels triangles adjacents.
- Per a cada aresta, la seva quàdrica és la suma de les dues dels seus vèrtexs.
- La quàdrica de cada aresta permet calcular el punt òptim (punt "verd"): és el punt de distancia mínima quadràtica a tots els plans que han intervingut a la quàdrica de l'aresta
- L'error associat a cada aresta és la distancia d'ella al seu punt òptim.
- A cada iteració s'escull com aresta a colapsar la que té error mínim

3.3.4 Altres metòdes

- Agrupament o clustering de vèrtexs (Rossignac I Borell, 1993)

- Mètodes basats en models volumètrics, que calculen un model de voxels i després reconstrueixen una nova malla de triangles interior al conjunt de vòxels que conteníen superficie original.

  En aquest cas, l'avantatge és que es pot garantir una cota d'error respecte la superficie inicial, cosa que no es pot fer en els mètodes incrementals com edge collapse, face removal o bé vèrtex removal.

3.4 Suavitzat de malles

Un dels algorismes més utilitzats és el de Gabriel Taubin, en que s'alternen iteracions de suavitzat amb operador Laplacià, amb iteracions que eviten l'encongiment de la superficie.

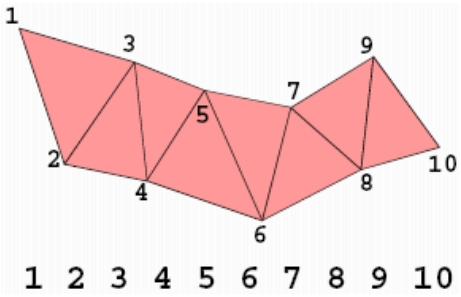
3.5 Edició de malles

Un algorisme simple per la modificacio de la forma d'una malla es pot basar en arrastrar els vertexs que queden dins d'una esfera:

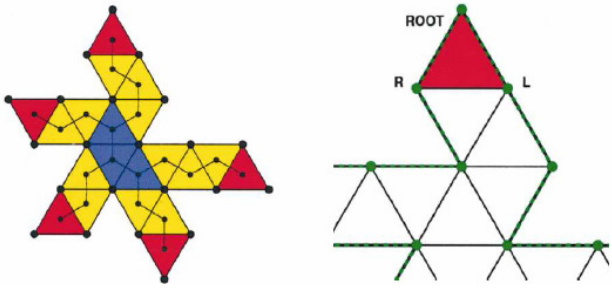
- L'usuari defineix un tamany d'esfera (radi r), i usa l'esfera per seleccionar un conjunt de vertexs, que son els que cauen dins de l'esfera
- En moure l'esfera, es mouen els punts que s'han seleccionat i que queden dins seu.
- Hi ha una segona esfera, exterior i concentrica a la anterior (radi R), per fora de la qual els vertexs de la malla no queden modificats durant l'edicio.
- Durant el proces d'edicio, els vertexs que queden dins de l'esfera petita es mouen junt amb l'esfera com si tot plegat fos un solid rigid.
- Un cop definides les dues regions a traves de les dues esferes i abans de començar a moure la malla, es calcula un valor o parametre per a cada un dels vertexs que estan dins de l'esfera gran i fora de l'esfera petita. Aquest parametre t per a un cert vertex v es simplement t=Dg/(Dg+Dp), on Dg es la minima distancia del vertex v als vertexs de la malla externs a l'esfera gran, i Dp es la minima distancia de v als vertexs de la malla interns a l'esfera petita.
- Els vertexs dins l'esfera gran i fora de la petita es mouen en funcio del valor del seu parametre, de manera que si s'aplica una certa transformacio geometrica TG a l'esfera petita, al vertex v se li aplica una transformacio geometrica t * TG.

3.6 Compressió de malles

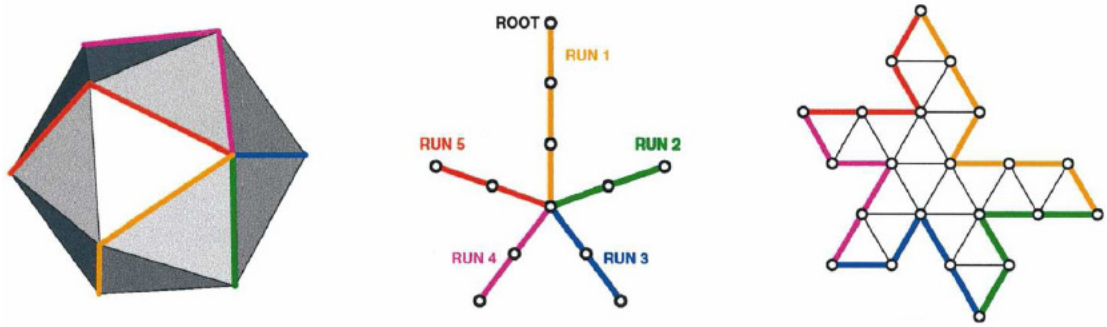
3.6.1 Triangle Strips



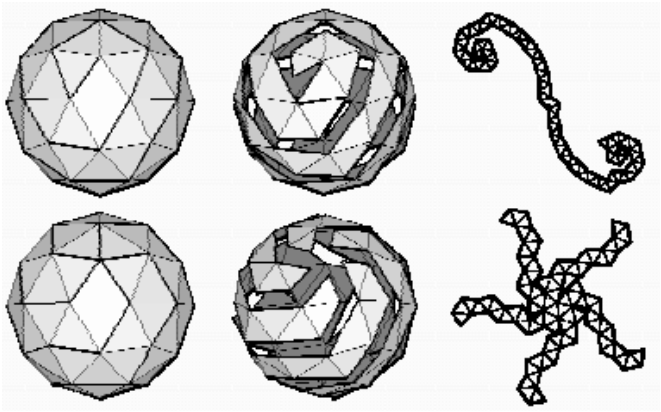
3.6.2 Triangle Spanning Tree (TST)



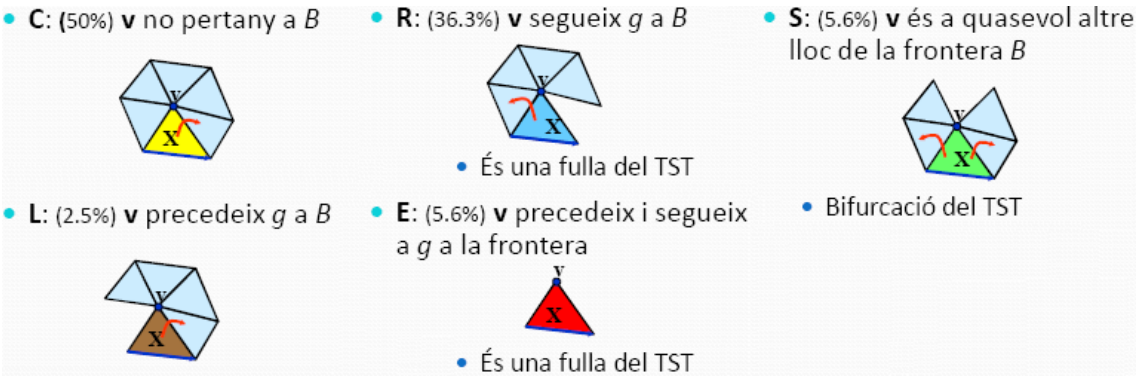
3.6.3 Vertex Spanning Tree (VST)



3.6.4 Topological Surgery (VST+TST)

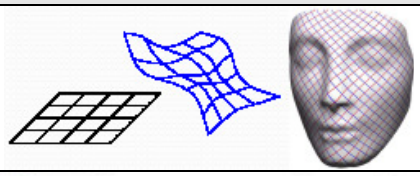
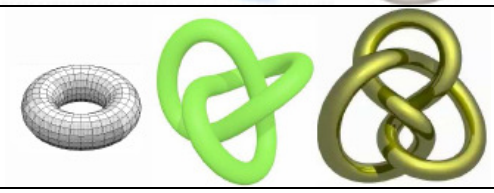


3.6.5 Edge Breaker



3.7 Parametrització i texturat de malles

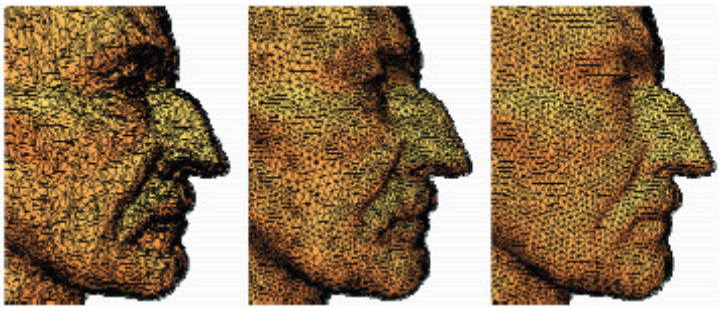
3.7.1 Superfícies topològicament equivalents (homeomòrfiques)

	Exemple	Homotópiques
a R^2		Si
a una esfera		Si
a un torus		Si
		No

3.7.2 Aplicacions

(a) *Texturació*

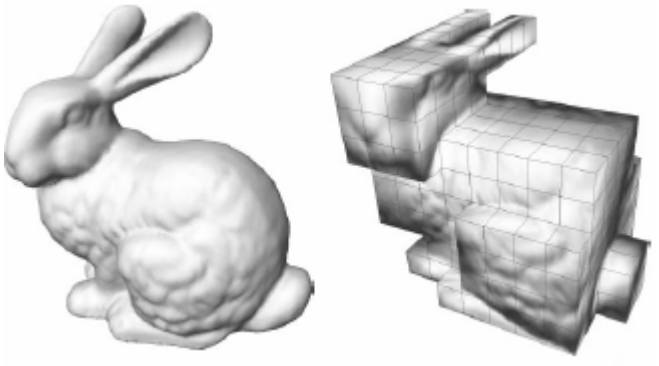
(b) *Morphing*



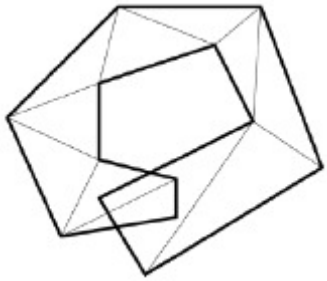
(c) *Remallat*

3.7.3 Tipus de parametritzacions

(a) *Planes*

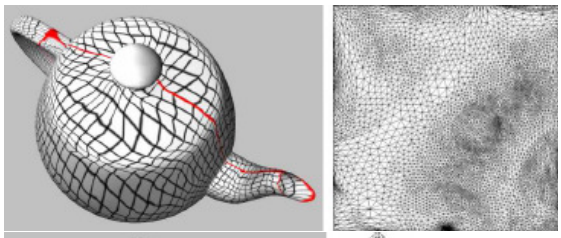
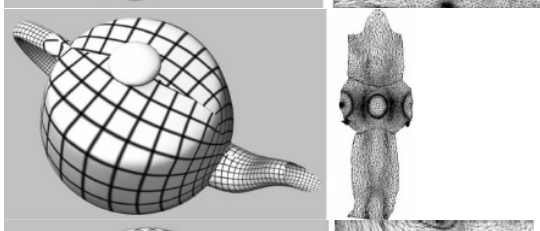
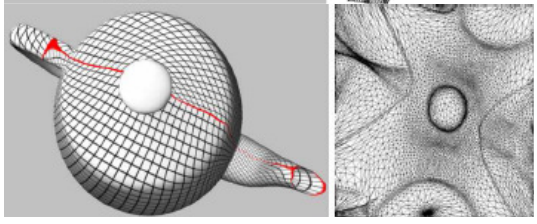


(b) *Sobre policubs*

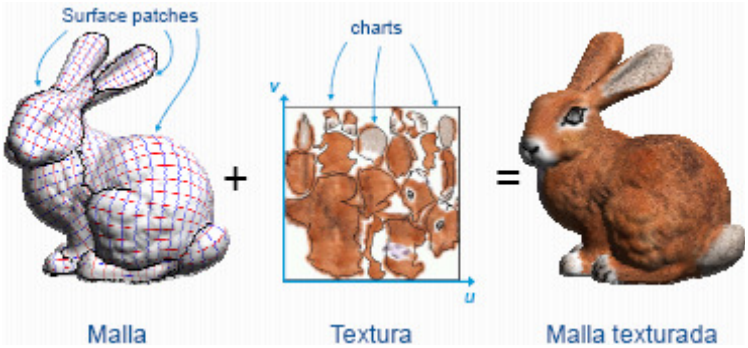


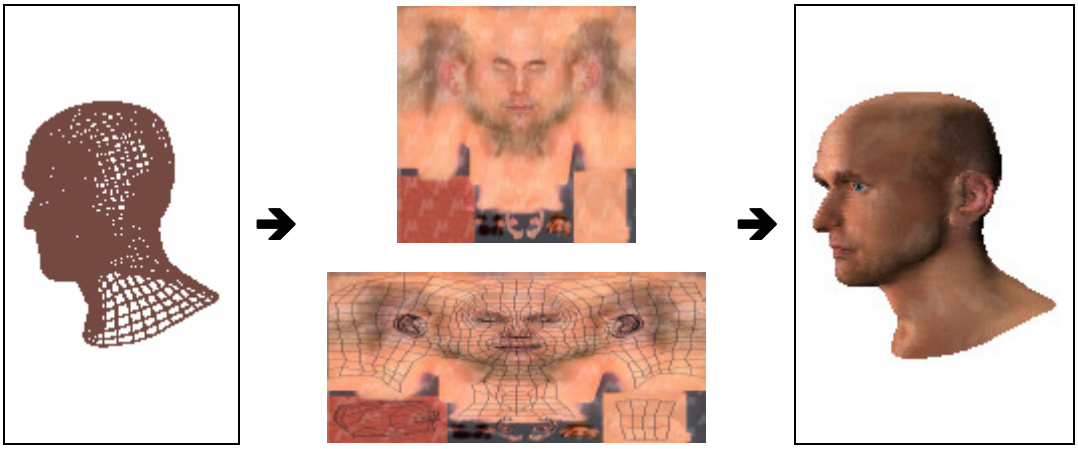
(c) *No bijectiva*

3.7.4 Evaluació visual de la distorsió

Baricèntrica de Tutte No minimitza ni distorsió d'angle ni d'àrea	
Conformal Minimitza la distorsió angular	
Equiàrea Minimitza la distorsió d'àrea	

3.7.5 Atles de textures





3.7.6 PolyCube-Maps

Transparències *Tema_3.7_PolyCubeMaps.pdf*

4 Realitat Virtual

La Realitat Virtual (RV) és una forma avançada d'inspeccionar i interaccionar amb els models, que intenta maximitzar la percepció de presència de l'usuari dins l'entorn virtual tot usant tècniques **d'immersió sensorial** i **d'interacció directa** i implícita.

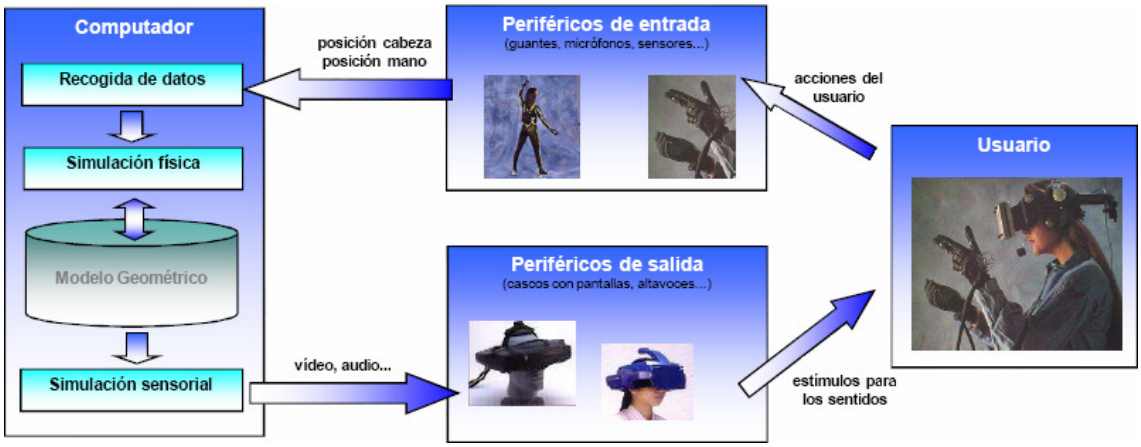
4.1 Introducció

Simulació	Recrea un món virtual que només existeix com model digital dins del computador • Simulació interactiva vs Animació • Passivitat, guió previ • Improvització, resposta: temps real • Representació geomètrica 3D • Representació de l'aparència • Algoritmes de visualització realista • Algoritmes de gestió de memòria • Models multiresolució • Capacitat de "zoom" • Preprocés de visibilitat
Interacció implícita	El sistema captura la voluntat de l'usuari implícita en els seus moviments naturals • Gestos, moviments del cap vs Interacció amb el mouse • Interacció, selecció: movements d'agafar amb la mà / dit, etc • Transparència dels dispositius i del computador • Percepció de la interacció directa amb els objectes • Finestra al model vs incorporació a l'entorn virtual
Inmersió sensorial	Desconexió dels sentits del món real, conexió a l'entorn virtual • Immersió visual: els objectes existeixen amb indep del disp de visualització • Visió estereoscòpica. Sensació de presència a l'espai • Immersió acústica • Immersió tàctil • Immersió de moviment: acceleracions • Olfacte, gust

4.2 Arquitectura d'un sistema de RV

Els sistemes de Realitat Virtual es composen de:

- Un computador que corre un programa de simulació, inspecció i/o interacció
- El model de l'escena virtual
- Dispositius de sortida (efectors)
- Dispositius d'entrada (sensors)



4.3 Visió Estereoscòpica en RV

El principal objectiu de la immersió en un sistema de realitat virtual, és aconseguir la immersió visual. Com que els humans tenim percepció estereoscòpica de l'entorn a través del nostre sentit de la vista, un dels elements esencials en tot sistema de realitat virtual són els sistemes que generen aquesta percepció estèreo. En un sistema de RV els objetes no es veuen a la pantalla, s'han de percebre en 3D, flotant a l'aire. Per això és important coneixer:

- Els fonaments de la visió estereoscòpica
- Les tècniques per a la codificació dels parells estereoscòpics

Transparències *Tema_4.3_Stereo_I.pdf*

Transparències *Tema_4.3_Stereo_II.pdf*

4.4 Dispositius de RV

Tal com ja hem comentat, ells perifèrics o dispositius d'un sistema de realitat virtual es classifiquen segons el sentit de la informació entre el participant i la màquina.

Els dispositius d'entrada, també anomenats sensors, capturen les accions del participant (per exemple moviments del cap) i envien aquesta informació al computador encarregat de dur a terme la simulació.

Els dispositius de sortida, també anomenats efectors, generen els estímuls necessaris pels sentits del participant, traduint en imatges, sons, etc. els senyals de vídeo, àudio, etc. que reben del computador.

- Dispositius efectors
- Dispositius sensors
- Dispositius de realimentació de força (ó hàptics)

Transparències *Tema_4.4_Dispositius_Entrada.pdf*

Transparències *Tema_4.4_Dispositius_Sortida.pdf*

4.5 Aplicacions

• Medicina:
– Simulació quirúrgica
– Tele-medicina
– Entrenament
– Rehabilitació neurològica

• Disseny mecànic:
– Ensamblatge de peces

• Entreteniment:
– Pintar en 3D
– Animació de personatges

• Visualització científica:
– Anàlisi de dades geofísiques
– Manipulació molecular

5 Animació

Les tècniques d'animació ens mostren canvis d'una escena en base a un guió preestablert. En general, el guió especifica com es transforma l'escena i com es mouen, canvien de color i es deformen els seus objectes.

Per a una introducció general al tema, llegiu el capítol corresponent a Animació del CD interactiu de Gràfics de l'assignatura de VIG.

5.1 Animació basada en esquelets

En la versió més simple, l'usuari especifica com es modifica la posició de cada tram de l'esquelet (que té molts menys punts que la malla de l'objecte que volem deformar) a cada una de les imatges clau o "key-frames" de l'animació. Ara, per a cada frame intermig, es calcula la posició de tots els trams de l'esquelet per interpolació, i finalment es calcula la posició de cada un dels vèrtexs de la malla de triangles de l'objecte que es deforma, a partir d'aquestes posicions de tots els trams de l'esquelet. Serveix per animar humanoids, animals etc. Per evitar errors durant l'animació, cal associar els vèrtexs de la malla de triangles que són propers a les unions entre trams de l'esquelet a més d'un tram d'esquelet.

5.2 Models deformables

Aquests mètodes intenten simular la deformació d'objectes tous que no tenen esquelet: òrgans i teixits humans, pilotes que xoquen i es deformen, vestits i robes, etc

Els mètodes més intuitius (i empírics) per a la simulació d'objectes deformables són els basats en la deformacio de l'espai. En aquest cas tenim una malla de control (que habitualment és regular i cúbica) que engloba l'objecte. La malla actúa com si fós un exoesquelet del nostre objecte a deformar. En la versió més simple, l'usuari especifica com es deforma aquesta malla de control (que té molts menys punts que la malla de l'objecte que volem deformar) a cada una de les imatges clau o "key-frames" de l'animació. Ara, per a cada frame intermig, es calcula la posició de tots els vèrtexs de la malla per interpolació, i finalment es calcula la posició de cada un dels vèrtexs de la malla de triangles de l'objecte que es deforma, a partir d'aquestes posicions dels vèrtexs de la malla de control. En un preprocés, es calcula per a cada vèrtex de la malla a quina cel.la de la malla de control pertany i quines són les seves coordenades respecte la malla de control (aquestes coordenades poden ser linials, coordenades baricèntriques basades en una descomposició en tetraedres dels cubs de la malla original, o bé basades en interpolació amb Splines, com fan en Sederberg o la Sabine Coquillart). A cada frame, un cop tenim la posició dels vèrtexs de la malla de control, només cal, per a cada vèrtex de la nostra malla de triangles, calcular la nova posició espaial tal que doni, respecte les noves posicions dels vèrtexs de control, les mateixes coordenades que s'havien calculat en el preprocés.

El problema és com deformem la malla de control, si no volem haver d'especificar, tal com deiem al paràgraf anterior, la posició de cada punt de la malla a cada key-frame. Els sistemes basats en masses i molles donen simulacions força realistes i eficients, són un bon compromís entre el realisme de les simulacions i la interactivitat. La malla de control es deforma seguint la dinàmica d'una malla de masses i molles. Les molles tenen una posició de repós que correspón a la posició estable de la malla de control. Quan estem a un estat d'equilibri, la suma de forces que exerceixen sobre un determinat vèrtex de la malla totes les molles que hi conflueixen, ha de ser cero. Si apliquem alguna força per deformar la malla, la suma de forces deixa de ser cero i aquesta es deforma. L'altra possibilitat és que l'usuari (amb la interfície) mogui o estiri algun dels vèrtexs de la malla. En aquest cas està estirant algunes de les molles amb la qual cosa la suma de forces torna a ser no nul.la i la malla es deforma. Una de les configuracions més usades per a la malla de control és la d'una malla cúbica uniforme, en que cada node o vèrtex de la malla té 18 molles que es conecten a:

- Els 6 vèrtexs que són veins seus en els dos sentits en cada una de les tres direccions coordenades
- Els 4 vèrtexs que són veins seus en diagonal, dins d'un tall de la malla que passa pel vèrtex i que és paral.lela al pla coordenat XY
- Els 4 vèrtexs que són veins seus en diagonal, dins d'un tall de la malla que passa pel vèrtex i que és paral.lela al pla coordenat XZ
- Els 4 vèrtexs que són veins seus en diagonal, dins d'un tall de la malla que passa pel vèrtex i que és paral.lela al pla coordenat YZ

A cada iteració, es calcula la suma de forces produides per les 18 molles a cada vèrtex de la malla (tenint en compte la llei linial de Hook que dóna la força d'una molla que s'ha estirat o comprimit), i es mou cada vèrtex proporcionalment a la seva força resultant. En alguns cassos es fa més d'una iteració d'aquest procés.

Si volem un comportament encara més realista, cal anar a mètodes basats en la física. Un dels més coneguts i usats és el mètode dels elements finits. El resultat de la simulació és molt realista, però pot ser força lent; la consequència és que en general aquest mètode només és aplicable quan l'animació es genera "off line". La tècnica es basa en dividir els objectes a animar en elements (si es vol una animació realista, els elements han de ser suficientment petits i cobrir tot el volum de l'objecte o dels obejctes a animar). Es calculen les deformacions de cada node de la malla, a partir de les forces aplicades. El càlcul de les deformacions (desplaçaments) es fa en base a resoldre un gran sistema d'equacions a cada frame.